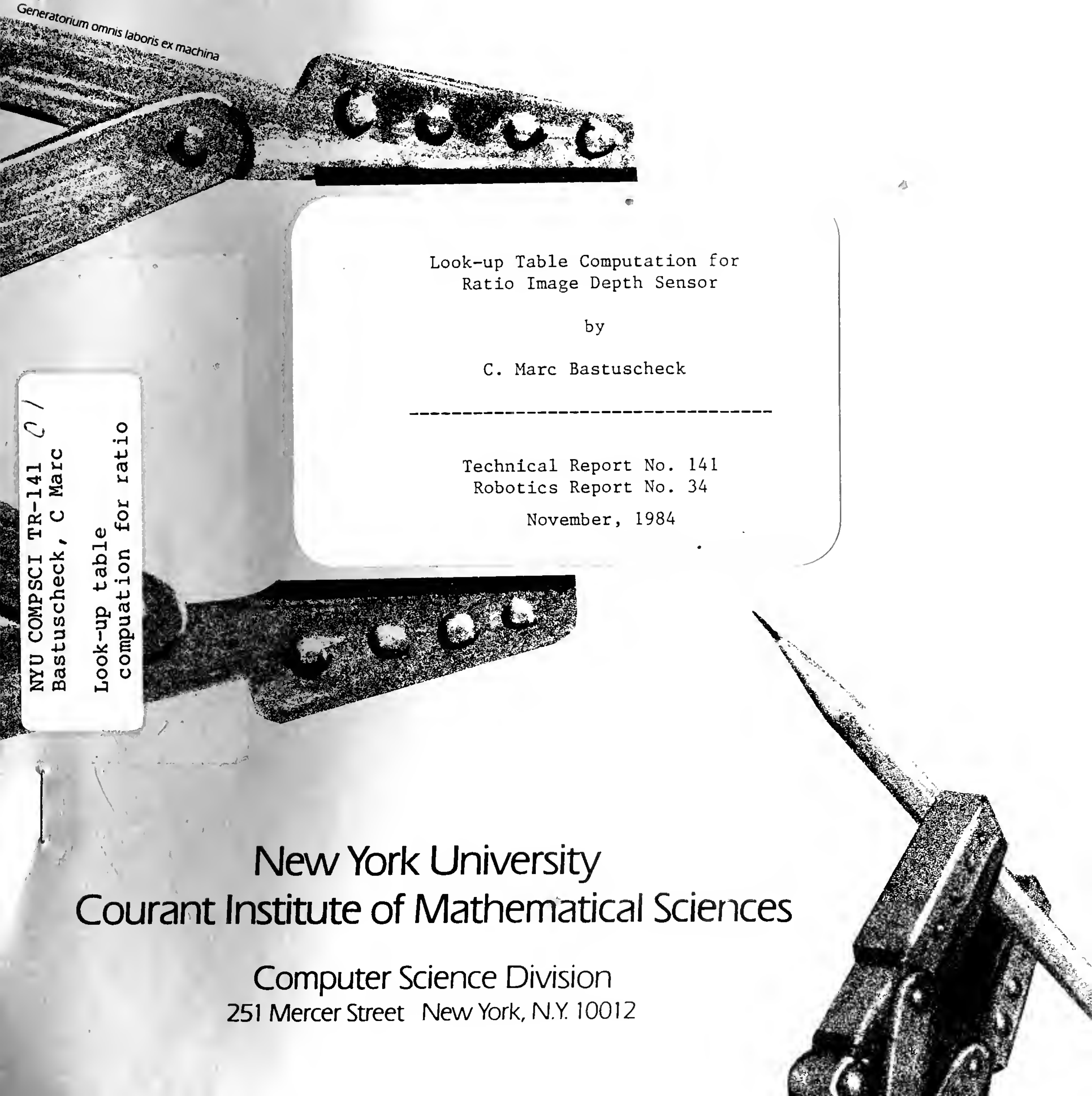
Robotics Research Technical Report

Generatorium omnis laboris ex machina



Look-up Table Computation for
Ratio Image Depth Sensor

by

C. Marc Bastuscheck

Technical Report No. 141
Robotics Report No. 34

November, 1984

New York University
Courant Institute of Mathematical Sciences

Computer Science Division
251 Mercer Street New York, N.Y. 10012

Look-up Table Computation for Ratio Image Depth Sensor

by

C. Marc Bastuscheck

Technical Report No. 141
Robotics Report No. 34

November, 1984

Look-up Table Computation for Ratio Image Depth Sensor

C. Marc Bastuscheck

Computer Science Department
Courant Institute of Mathematical Sciences
New York University

ABSTRACT: The theoretical basis of a method of computing the depth image from the ratio of two intensity images is described. One implementation of this method forms a 512x512 pixel depth image in 30 seconds, 25 of which are used to digitize images, with an average deviation of about 0.6% of the depth of the workspace, a resolution which is sufficient for experiments in shape analysis and object recognition. This report is written to serve as an introduction of the method to new users of the depth sensor.

Introduction:

The spatial coordinates (x,y,z) of a point on the surface of an object can be obtained from the intersection of the appropriate ray from a camera (indexed *e.g.* by the pixel location in the image) with a vertical plane from a projector which is displaced horizontally from the camera. Fig. 1 sketches a two dimensional reduction of this process. A number of plane of light triangulation schemes have been implemented, starting in the early 1970's [1,2] and continuing through recent commercial products [3,4].

Data from plane of light sensors is usually analyzed in its natural form of lines (from the intersection of the planes of light with objects) [5-7]. For example, planar object surfaces show parallel, evenly spaced straight line segments, the ends of line segments denote edges which can be fit by straight lines or curves, etc. From this information features of interest (holes, edges) or entire objects can be identified. However, data for an entire scene can only be built up from a number of sequentially processed images of planes of light.

Recently a method of acquiring depth information from an entire scene at once was proposed [8] which uses the ratio of two images formed with the work area illuminated sequentially by two beams of light having different intensity variations across them. The resulting "ratio image" provides a continuous coding of the projected beam which allows the identification of an arbitrarily large number of planes of light in the workspace. Experiments with a preliminary implementation of this method [9] established that the assumptions on which the ratio image depth sensor is based (reproducible lighting conditions, linear response of the camera to light) could be achieved technically. The actual computation of depth values in that preliminary work was done by linear interpolation between ratio images of a plane placed first at the front of the workspace, then at the back, a procedure which produced depth images characterized by systematic distortion from the inappropriate use of linear interpolation and by low resolution in depth due to magnifying the small differences of large numbers. A quadratic interpolation method of calculation using three calibration planes reduced systematic distortions but at a cost of increased "noise" in the

Work on this paper has been supported in part by Office of Naval Research Grant N00014-82-K-0381, and by grants from the Digital Equipment Corporation, the Sloan Foundation, the System Development Foundation, the IBM Corporation, and by National Science Foundation CER Grant No. DCR-8320085.

depth images.

An observation by Robert Hummel led to the development of the present look-up table method of computing the depth image of a work scene from the ratio image of the scene. Since the depth image is merely the ratio image run through a look-up table plus small corrections, the resolution of the depth image is the same as that of the ratio image, which can be 8 or 9 bits. The method produces depth images entirely free of systematic distortions. Since look-up tables can be loaded into the VICOM, all processing except the off-line preparation of the look-up table is done quickly on this machine.

Section 2 of this report details the derivation of the look-up table method. Section 3 describes the implementation of the method on the VICOM. Section 4 describes calibration methods used to gather data necessary to construct the ratio-to-depth look-up table. Section 5 describes in broad terms how the look-up table is calculated in the present implementation; the actual code (in C) is contained in Appendix A. Section 6 contains images formed using this look-up table method and a short discussion of the present application of the method.

2. Mathematical Basis of the Look-up Table Method

Figure 2 sketches the geometrical relations which underlie the look-up table method. We consider for simplicity the two dimensional case, which extends trivially to three dimensions when vertical planes of light are used in place of the 2D rays from the projector. The camera lies at the origin of the coordinate system and looks along the z-axis, and rays from the projector diverge from the point (x_0,z_0). An arbitrary point in the workspace (x,z) is determined by the intersection of the ray from the camera at angle θ to the z-axis and the ray from the projector which intersects the z-axis at the point $(0,d)$.

For convenience in the following argument we draw line segments from (x,z) and (x_0,z_0) perpendicular to the z-axis. Then by similar triangles it is clear that

$$\frac{z-d}{x} = \frac{z_0-d}{x_0}. \tag{1}$$

In this expression the quantities are considered to be coordinate values, and thus can be negative, as *e.g.* x_0 is here. This can be rewritten as

$$z-\frac{z_0-d}{x_0}x = d$$

or

$$z[1 - \frac{z_0-d}{x_0}\frac{x}{z}] = d$$

or,

$$z = \frac{d}{1 - \frac{z_0-d}{x_0}\tan\theta} \tag{2}$$

where x/z has been written as $\tan\theta$.

The key observation at this point is that if the ratio varies strictly monotonically across the beam, every ratio value corresponds to a unique value of d on the z-axis. Thus within the beam it is possible to define a function $d(\rho)$ which maps the observed ratio value ρ into a distance on the z-axis. For regions of the image sufficiently close to the center vertical line, $\tan\theta$ is very small, and this $d(\rho)$ function (which is easily implemented as a look-up table) converts the ratio observed at any pixel to the depth (z coordinate) of the part of an object imaged at that pixel.

For purposes of translating ratio images to depth images we can simply replace the d of eq. 2 with $d(\rho)$.

3. The VICOM Implementation

The VICOM [10] is a special purpose computer designed for image processing. It has a firmware operating system to implement low level commands, and serial and parallel connections to a host computer (a VAX 750 running Unix 4.2 bsd) to allow the passing of commands and images between the machines. The shell vsh [11] provides a very convenient software interface.

The basic data unit of the VICOM is the 512x512x12 bit memory plane. Programs written to run on the VICOM (chain files of VICOM commands, or "vsh macros") can be thought of as being written in a machine language with these memory planes as registers; individual pixels cannot be addressed without a great deal of effort. The basic processes of addition, subtraction, multiplication, and division work on corresponding pixels of images: $A_{i,j} + B_{i,j} = D_{i,j}$ for all i,j in the images A, B, and D. In addition there is the ability to run an image through a look-up table, and there are numerous functions useful in image processing. The availability of only 12 bits during all processing can limit attainable depth resolution unless special effort is made in writing command files.

To improve depth resolution eq. 2 should be written so the available bits span the workspace exactly. For example, suppose that all depths are to be measured from a plane perpendicular to the z-axis at d_0, and let $z = Z + d_0$ relate the new depths Z to the old depths z. Making this substitution in eq. 2 results in

$$Z = \frac{\delta(\rho) + d_0 C}{1 - C}, \tag{3}$$

where $\delta(\rho)$ is now the look-up table, and $C = \frac{Z_0 - \delta(\rho)}{x_0} \tan\theta$ is a small correction. In this Z representation all 2047 positive numbers (from 0 to 1.0 on the VICOM) can be used to represent depths within the imaged area. This places a limit of resolution of about 0.5 mm for a work area of 1 meter extent, beyond the present resolution of the ratio image. The depth values generated by the sensor are converted to physical dimensions by multiplying by the "computer unit", which is the physical distance corresponding to 1.0 in the VICOM units. The computer unit is created by the program which makes the look-up table.

Since all processing must be performed on an image as a whole, it may not be immediately clear how eq. 3 can be implemented, since $\tan\theta$ is spatially varying. However, one need merely create an image for each quantity on the right hand side of eq. 3. The look-up table can be used to pass constants such as d_0 and x_0 to the program running on the VICOM. However, the correction C must be approached carefully, since Z_0 and x_0 are both larger than a computer unit, and thus do not fit in the VICOM memory. C is formed by combining the quantities $\frac{Z_0}{x_0}$, $1/x_0$, and d_0, each of which is constrained to be smaller than one when the look-up table is made. These values are passed in the ratio-to-depth look-up table using ratio indices outside the normal range.

To make a depth image it is necessary to make calibration measurements, create the look-up table, and acquire a suitable ratio image of the scene. The ratio image is run through the look-up table to give the raw depth image. Three constant images are made with appropriate index values and run through the look-up table to give $\frac{Z_0}{x_0}$, $1/x_0$, and d_0 images. The $\tan\theta$ image must in fact by created by the host computer (in the present experiments a straight line approximation is used), but can be passed as a small 16x512 image to the VICOM and stored in the same large plane as the look-up table. This small image is copied several times to form a full size image. Another image is made for the constant 1. At this point each quantity of Eq. 3 exists as an image, and VICOM commands can be used to compute the fully correct depth image.

4. Calibration Procedures

The information necessary to construct the look-up table is simply the value of the ratio at the z-axis where it is illuminated by the projector. If the projector optics and the transmissivity of the filters were known, then the $d(\rho)$ function could be calculated for any desired geometry. It is easier, however, to measure the projected ratios in a way which is more directly applicable to the calibration procedure.

A direct measurement of $d(\rho)$ is difficult since, as shown in Fig. 3, the points of interest are all imaged on the same point of the two dimensional camera, or to a vertical line in three dimensions. However, the projected ratios can be determined by moving the screen to a more convenient angle to the camera. Measurements would be made (perhaps with the screen in several different locations) and the ratios observed on the screen could be computationally mapped to give $d(\rho)$ on the z-axis.

The method which has been used in generating the look-up tables for the present experiments in fact does not remap the ratio values. The configuration shown in Fig. 3 is altered by rotating the projector and the screen 90° about the central point of the workspace, defined as the intersection of the central ray from the camera and the central ray from the projector. The projector must retain the same angle to the screen in this calibration set-up as to the central camera ray in the measuring geometry. In this way 512 ratio values corresponding to the 512 horizontal pixels are measured at once. Additional tolerance to local ratio deviations, due perhaps to dirt on the filters or in the optics, is obtained by averaging vertical columns, since each column should have the same ratio value. In the present implementation of this method 16 rows spaced along the central half of the image are copied to the VAX for averaging and forming the look-up table.

One difficulty with this method is that some areas which intuitively should be part of the working area of the depth sensor lie outside the region illuminated by the projector. Fig. 4 illustrates the problem: the shaded corners of the square are not part of the working area. This problem is overcome by making the calibration measurements with the projector at a distance chosen so the beam conveniently fills the field of view of the camera, then setting up for depth measurements with the projector at a greater distance or with the zoom lens adjusted to give a larger beam. The calibration process thus consists of two procedures: first, the ratio is recorded with the projector at the chosen angle to the screen (equipment in the calibration arrangement), and second, the coordinates of the projector and the distance to the center of the workspace (camera is at the origin) are recorded (with equipment in the measuring location). A full description of the process follows:

1) Place all apparatus in the measuring position. Bear in mind that depth accuracy increases and quality of illumination decreases as the angle between the projector beam and the camera axis approaches 90°; 45° has worked well as a compromise.

2) Use the "CAM" command to place a screen along the camera axis, defined by a vertical line on the center of the monitor. A screen of matte white Formica mounted on heavy plywood combines flatness and good properties of reflection.

3) Using a slide with markings, find the center of the projector beam on this screen. Record the angle of the projector to the screen.

4) Rotate the screen 90° so it is square to the camera (use perpendicularity device).

5) Place the projector at the same angle to the screen as in step 3 and at such a distance that the useful portion of the beam fills the field of view of the camera. Adjust the equipment so the center of the beam falls on the screen at the center of the camera field of view. If the angle of the projector to the screen is changed, record the new angle for use in setting up the projector for measurements. Adjust the projector

so the projected slides are aligned with the vertical.

6) In this calibration geometry, record the width of index marks projected on the screen using a specially prepared slide. A grid slide is convenient for this and for adjusting the projector focus. (In these measurements the projector and camera have been focussed on the center of the workspace).

7) Make a ratio image using the chosen filters, and copy 16 lines of the image to the VAX as "raw.img" for processing. Multiple digitizations should be made and averaged to reduce the effects of digization noise and temporal instability of the projector. Typically 8 successive digitizations are made, which results in intensity images having about 10 good bits. Check the ratio image to make sure the lines of constant ratio are vertical and well defined.

8) Place the projector in the depth measuring position. At this time the work space is defined by the placement of the projector and the size of the projector beam (which can be varied using the zoom adjustment). The only constraint is that the angle between the center of the projector beam and the camera axis be the same as the angle between the plane of the screen and the projector beam in the calibration geometry. Check that planes of constant ratio are vertical. With the screen along the camera axis record the width between the projected index marks as in step 6. Record the distance to the center of the workspace from the camera and the x and z coordinates of the projector. (Unlike the pinhole camera and projector drawn in the figures, real optical systems have no single point through which all rays pass. Experiments to determine the effective point of divergence have placed this point near the front element of the lens systems, and nearly any point used consistently seems to work.)

9) Run delcalc*, the program which makes the look-up table. Enter the projector location and the distance to the center of the workspace. When prompted for the "grid width ratio" enter *MW/CW*, where *MW* and *CW* are the widths between calibration marks observed in the measuring and calibration positions respectively.

At this point the VAX creates the file delta.lut which is used in the chain files to make depth measurements using a pair of filters. This file can be used to make depth images at any time in the future by using the same filters and placing the projector relative to the camera exactly as it was when the look-up table was made. Note that this method results in a work area which is symmetrical in depth about the midpoint of the beam, *i.e.* some usable ratio and depth pairs have not been incorporated into the look-up table.

5. Overview of the Program delcalc.c

Commented code for the program which creates the look-up table is reproduced in Appendix A. The code has the following main functions:

Input. This prompts for the projector position, distance to the screen and the grid width ratio. The last two are needed to construct the look-up table; the projector position must be input to ensure that later computations on the VICOM will not overflow. The distances can be measured using any units, but centimeters are a good choice.

Computer unit. The distance corresponding to the difference between the closest and farthest points on the z axis which can be recorded with the equipment in measuring position is calculated. Since this distance is to be mapped into depth values between 0 and 1, this distance is the computer unit. However, if the distance from the camera to the center of the workspace is larger than this the computer unit is made larger (and the range of depth values becomes correspondingly smaller). If the chosen measuring geometry is inappropriate for the calculations an error message is generated and the program exits.

Averaging. The first processing of the raw ratio values, indexed by horizontal pixel number, is to average all 16 values for each horizontal position. This reduces the effects of noise and makes the look-up table representative of a larger region of the image.

Smoothing. Even with averaging the possibility exists that noise will result in locally non-monotonic behavior in the averaged points. To encourage monotonic ratios this routine places at each point the average value of the 7 adjoining pixels centered on that point. While this process does not guarantee a monotonic ratio sequence, non-monotonic behavior has not yet been observed.

Inversion. The processing so far has dealt with ratio values indexed by pixel location, which corresponds to the depth. The maketable routine turns the relation around, and creates a file of depth values indexed by ratio. 512 evenly spaced depth values are computed between the far and near points, and each depth value is placed in the slot of its corresponding ratio. Small distortions which exist in the camera optics and detector have not been taken into account in this routine; this results in a systematic distortion on the order of 0.5% of the depth of the workspace for extreme depths.

Interpolation. At this point there are only 512 ratio indices filled of the 2047 positive indices available, so the unfilled slots are filled by linear interpolation between the nearest filled neighbors. Depth values indexed by ratios less than or greater than the filled areas are set to zero.

Parameter passing. The locations of the projector and the center of the work space are passed in a form which can be used in the VICOM calculations, namely as Z_0/x_0, d_0, and $1/x_0$ with the values in computer units. These are passed in table locations 4090, 4091, and 4092 (FFA, FFB, and FFC hex). For reference, the number of centimeters in a computer unit is passed, multiplied by 10 to preserve a decimal place, in 4093 (FFD hex).

Writing. When complete, the table of 4096 short integers is written to disk in a format which will be loaded correctly by the vsh interfacing routines. The file is called delta.lut, and overwrites any previous file of that name.

6. Representative Results

To illustrate the capabilities of this implementation of the ratio image depth sensor a simple 'work scene' was constructed having a sphere and letters cut from paper at different distances from a back plane. Fig. 5 is an intensity image of this scene as viewed by the tv camera. Some notable features of this intensity image are the dark paper taped to the white screen at the back of the scene, the grain in the wood block at the lower left, and the way the letters blend into the background. (The shadows suggest to the human viewer how the projector is offset from the camera.)

Fig. 6 is a photograph of the depth image of this scene, a grey scale image where larger numbers (brighter) are farther from the camera. The back screen is uniformly bright because the dark paper is at the same depth as the white screen. Similarly the dark grain of the wood block is not visible in the depth image. The letters DEPTH advance towards the camera. The shadow regions are tagged black to identify them; there is of course no information available to the depth sensor in shadow regions. This depth image was formed in 30 seconds, of which 25 seconds were used in acquiring the ratio image (8 digitizations of each intensity image were made to reduce noise by averaging) and 5 seconds were needed for computation.

By threshholding the depth image regions of the image having a narrow range of values can be selectively displayed. In Fig. 7, for example, several slices of the depth values are reproduced. Each of these slices is about two cm thick. No attempt was made to smooth the values of the surfaces by spatial averaging. The P and T differ by about 5 cm in depth. The overall depth of imaged area was about 80 cm.

Unwanted variation (noise) in the intensity images translates into error in the derived depth images. Analysis of the error in these depth images and in others made with the same technique showed a random variation which resulted in average deviations for groups of 6 pixels of between 0.2 and 0.5 cm (average 0.3 cm) in a bright region, and between 0.45 and 1.05 cm (average 0.8 cm) in a region having four times smaller reflectivity. These results are consistent with earlier uncertainty estimates [9]. There was also a coherent variation, where the entire image became closer or farther, having an average deviation of similar size; the effect is probably related to small temporal variations in the intensity of the slide projector lamp. Even with this noise the relative change of location of a dark surface by 0.3 cm was clearly observed when 4 adjacent pixels of a subsampled image were averaged. The pixel-to-pixel variation in depth makes a surface appear very uneven at small scales, and thus gradient operators of the sort often used for edge detectors (3x3 convolution boxes) give very noisy results on depth surfaces. Jump edges of objects (where an object is separated from the background) are of course very sharply determined by edge detectors.

Besides the deviations due to noise and intensity variation there are small systematic disturbances to the depth image. The most important of these is that of secondary illumination, where light is reflected from an object in the beam to a visible surface. Instances of this are visible in Fig. 6, both in the scattered points in the shadow of the P, and also on the upper right limb of the sphere, which should be in shadow. The effect is usually very small, except where the direct illumination is very weak or nonexistent.

7. Summary

The present implementation of the depth sensor acquires a 512x512 depth image in 30 seconds, 25 of which are used to acquire the ratio image. The result is a depth image with an average deviation of approximately ± 0.5 cm over a working area of 80 cm centered 80 cm from the camera. The projector is 70 cm behind and 150 cm to the left of the camera in these experiments. The calibration measurements take about 15 minutes, most of the time being spent in setting up and making geometrical measurements. Construction of the look-up table takes about 30 seconds (elapsed time, including data input) on the VAX.

This implementation of the depth sensor provides relatively good depth data quickly and easily, and is adequate for use in experiments in shape analysis and object recognition, since additional accuracy in fitting surfaces to depth data can be obtained by limited spatial averaging. In such studies the slight absolute displacements caused by the intensity fluctuations should not matter. If the absolute position of an object is important, as for instance when object coordinates are to be passed to a robot, these fluctations can be corrected by measuring depth relative to an object in a known location in the workspace. The look-up table calculations which are used in this implementation are also used in the sawtooth filter implementation which is currently under development.

References:

1. Y. Shirai and M. Suwa, Recognition of Polyhedrons with a Range Finder, Proc. 2nd IJCAI, pp. 80-87 (1971).

2. G. J. Agin and T. O. Binford, Computer Description of Curved Objects, Proc. 3rd IJCAI, pp. 629-640 (1973).

3. The White Scanner, Technical Arts Corp., PO Box 15390, Seattle, Washington 98115.

4. Robotic Vision Systems Inc., 425 Rabro Dr. East, Hauppage NY 11788.

5. K. Sugihara, Range Data Analysis Guided by a Junction Dictionary, Artificial Intelligence **12** , pp. 41-69 (1979).

6. M. Oshima and Y Shirai, Object Recognition using 3 Dimensional Information, Proc. 7th IJCAI, pp. 601-606 (1981).

7. R. C. Bolles, P. Horaud and M. J. Hannah, 3DPO: Three-Dimensional Parts Orientation System, Proc. 8th IJCAI, pp. 1116-1120 (1983).
8. J. T. Schwartz, Structured Light Sensors for 3-D Robot Vision, NYU Courant Institute Robotics Rpt. no. 8 (1983).
9. C. M. Bastuscheck and J. T. Schwartz, Preliminary Implementation of a Ratio Image Depth Sensor, NYU Courant Institute Robotics Rpt. no. 28 (1984).
10. Vicom Systems, Inc., 2307 Bering Drive, San Jose CA 95131.
11. D. Clark and R. Hummel, *VSH* Users' Manual: An Image Processing Environment, NYU Courant Institute Robotics Tech. Report (Oct. 1984).

```
/*  delcalc.c takes the file raw.img and constructs from
  it the lookup table  delta.lut  for use in deldepth.vc.

  Marc Bastuscheck  Oct 4, 1984
  coding of early programs: Nina Hachigian (July '84).

  Program based on Oct. 3 version of lutcalc.c, but modified to find a
  computer unit which uses nearly all possible positive 12 bit values to
  code depth in the working area of the sensor.  The look-up table
  inverts the depth-ratio file, and places Z0/x0, d0, and 1/x0 values
  at FFC, FFB, and FFA (hex) in the file to be loaded by the VICOM.
  These dimensions are in computer units; for reference, 10 x the cm
  equivalent of the computer unit is in FFD hex.                          */


#define abs(x)  ((x)<0? -(x): (x))

#include <stdio.h>
#include <iattr.h>

struct iattributes iattrs = {0,0,0,0,0,0,0,0,0,0,0,0 };

short outbuf[4096];  /*declare global buffers*/
float bufs[512];

main()
{
float z0, x0, d0, dscreen, tantheta, cunit, temp, near, far;
float readval();
char response;

tantheta = 0.2217;      /*  tan 12.5 deg, half angle of 25mm lens       */


/*                      parameter input section                         */

printf("Do you have a suitable raw.img?\n");
scanf("%c",&response);
if (response == 'n'|| response == 'N')
 {
   printf(" Goodby.\n");
   goto exit;
 }

printf("Enter the x coordinate of the projector in cm. Projector\n");
printf(" coordinates to the left of the camera are negative.\n");
x0= readval();

printf("Enter the z coordinate of the projector in cm's. Projector \n");
printf("coordinates behind the camera are negative.\n");
z0= readval();

printf("Enter the screen distance in cm's\n");
dscreen= readval();

printf("Enter the grid width ratio. (camera half angle 12.5 deg assumed)\n");
```

Appendix A

```
temp = readval();

tantheta = tantheta*temp;
temp = dscreen * tantheta;
d0 = dscreen - temp;
cunit = temp + temp;

/*  VICOM computation assumes that certain conditions pertain.  The
    following section ensures that the calibration being made will work.   */

if (d0 > cunit) cunit = d0;   /*  require d0 < = cunit                      */

z0 = z0 - d0;                 /*  means that z0 is actually Z0 from now on */
if (abs(z0/x0) > 1.0 )
  {
        printf("|Z0/x0| > 1 : change set up and try again.  Good-bye!\n");
        goto exit;
  }

if ( abs(x0) < cunit )
  {
        printf("|x0| too small: change set up and try again.  Good-bye!\n");
        goto exit;
  }


/*      Send relevant parameter values to the terminal                      */

printf("The computer unit is %f\n",cunit);
printf("Plane d0 is at %f\n",d0);
temp = temp/cunit;                            /* i.e. dscreen*tantheta/cunit */
near = temp -.5;
far = temp + .5;
printf("Depth values exist between %f and %f\n",temp-.5, temp+.5);

/*  Finally get to the business part of the program:                        */

getaverage();

smooth();

maketable(far,near);

/* outbuf now has a full set of distance values for expected ratio values
  Pass 1/x0, d0, Z0/x0 and 10*cunit (ref. only) using indices in the table */

d0 = d0/cunit;
x0 = x0/cunit;                      /* convert distances to computer units  */
z0 = z0/cunit;

if (x0 > 0.0)
        outbuf[4090] = ( 2047.0/x0 + .5);
   else outbuf[4090] = ( 2047.0/x0 + 4096.5);

outbuf[4091] = (2047.0*d0 + .5);
```

```
                                                        /*  reuse the variable   */
z0 = z0/x0;
if (z0 > 0.0)
        outbuf[4092] = ( z0 * 2047.0 + .5);
   else outbuf[4092] = ( z0 * 2047.0 + 4096.5);

outbuf[4093] = ( cunit*10.0 + .5 );     /*  Store it where it can be found   */

/*  4092 is FFC with z0/x0;    4091 is FFB with d0;  4090 is FFA with 1/x0 */

/*  The file is complete; write to disk in form VICOM wants to read.        */

writefile("delta.lut");
exit:
printf("That's all, Nina.\n");

}
/*     * * * * * * * * * * * * * * * * * * * * * * * * * * * *    READVAL  */

float readval()                /*   routine to check for errors during input  */

{
char garbage[10];
int i;
float val;

while ((i=scanf("%f",&val))!=1)
    {
     scanf("%s",garbage);
     printf("You have entered faulty data please try again\n");
    }
return(val);
}
/*     * * * * * * * * * * * * * * * * * * * * * * * * * * *  GETAVERAGE  */


/*  getaverage expects a file 16 rows x 512 columns, of 12 bit ratio values.
The function reads the file rav.img, averages the lines, and puts the final
(float) average in global variable bufa.                                   */

getaverage()

{
extern float bufa[];
short inbuf[512];
int imd,i,j,k;


imd = iopen("rav.img",0);
i = iattr(imd,&iattrs);

for (k=0; k<512; k++)                       /*  initialize bufa to all zeros   */
       bufa[k] = 0.0;

for (i=0; i<17; i++)
```

```
  {
       j = ireads(imd,inbuf,512);
       for (k=0; k<512; k++)
              bufa[k] = bufa[k] + inbuf[k]/16.0 ;
  }
}
/*     * * * * * * * * * * * * * * * * * * * * * * * * * * * *     SMOOTH  */

/*  smooth takes a globally available buffer bufa and places the average of
  7 values in the place of each middle value.  End 3 points not smooothed. */

smooth()

{
extern float bufa[];
float buft[512],tem[3];
int i,j,k;
float sum;

sum = 0;
for (i=0; i<7; i++)
        sum=sum+bufa[i];

for (j=3; j<508; j++)
  {
        buft[j] = sum/7.0;
        sum = sum + bufa[j+4] - bufa[j-3];
  }
buft[508] = sum/7.0;           /*   3 thru 508 of buft have smoothed values  */

for (i=3; i<509; i++)
    {
        bufa[i] = buft[i];
    }
return ;
}

/*     * * * * * * * * * * * * * * * * * * * * * * * * * * * *  MAKETABLE  */

/* This subroutine makes the ratio-to-depth look-up table.
  bufa is global 512 float ratio values smoothed and (I hope) monotonic.
  bufa's index represents 512 points from df to dn (far and near points).  */

maketable(df,dn)
float df,dn;

{
extern float bufa[];
extern short outbuf[];
short tembuf[512];
int i, entry, end, end1, end2;
short depth;

for ( i=0; i<4096; i++ )                          /*  initialize outbuf to zero */
```

```
        outbuf[i] = 0;

for ( i=3; i<508; i++ )
  {
        depth = ( ( df - ((df-dn)*i)/511.)  * 2047.0 +.5);
        entry = bufs[i] + .5 ;
        if (entry < 0)
                entry = 0;      /*  put all presumably bad entries at zero */
        outbuf[entry] = depth;
  }

outbuf[0] = 0;                                   /* now get rid of bad entry */
for (i = 2048; i<4096; i++ )
        if ((end=outbuf[i]) != 0)                /*   and check for others    */
                printf("depth at negative ratio observed\n");

/*  505 depth values are recorded for table entries indexed by the ratios
  available in the calibration data.  Now find the ends for interpolation. */

i=0;
end1=0;
while ((entry = outbuf[i]) == 0 )
        end1 = ++i ;                /*  exit with end1 at first occupied slot   */


i=2047;                         /* ignore negative part of look-up table here */
end2=2047;
while ((entry = outbuf[i]) == 0 )
        end2 = i-- ;              /*  exit with end2 after last occupied slot   */


/*  Fill in all table values between these ends by linear interpolation
    between nearest filled table values.                                     */

i=end1;                                 /*  start at first real occupied slot   */
while (i < end2 ) {
        while ((entry = outbuf[++i]) == 0 )
                ;
        end=i;                /*   when next full spot is found save i value */

        i = end1;                            /* end1 is previous good filled spot */
        if (end < end2)
        while ( ++i < end )                /*  interpolate between filled slots   */
            outbuf[i] = outbuf[end1] + (int)  ((outbuf[end] - outbuf[end1])*
                        ( (float)  (i - end1) / (float)  (end - end1)) +.5 );

        end1 = end;          /*                    save this filled slot as new base */
        i = end1;           /* and set up i to be as it was when we started */
                                              /* Continue until i reaches end2.  */
  }

/*  bufs now contains all legal depth values indexed by ratio values
    observed during the calibration.                                          */

return ;
}
```

```
/*     * * * * * * * * * * * * * * * * * * * * * * * * * * * * *  WRITEFILE  */

writefile(name)                               /*writes contents of buffer to file*/
char *name;
{

int i,imd;

imd= iopen(name,1);
i = iattr(imd,&iattrs);
i= iwrite(imd,outbuf,4096);
i= iclose(imd);
}
```

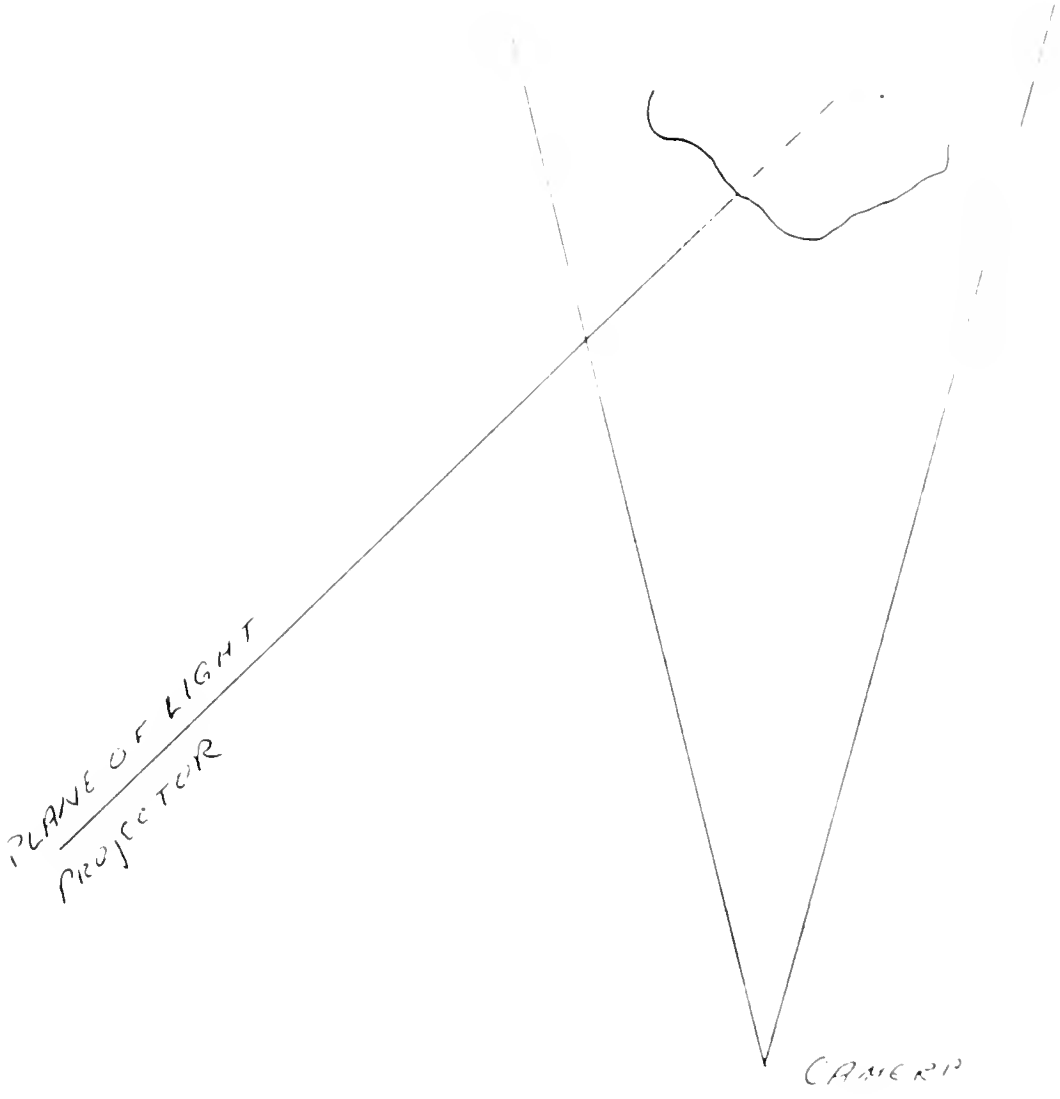


Fig. 1. Arrangement of equipment for plane of light triangulation.

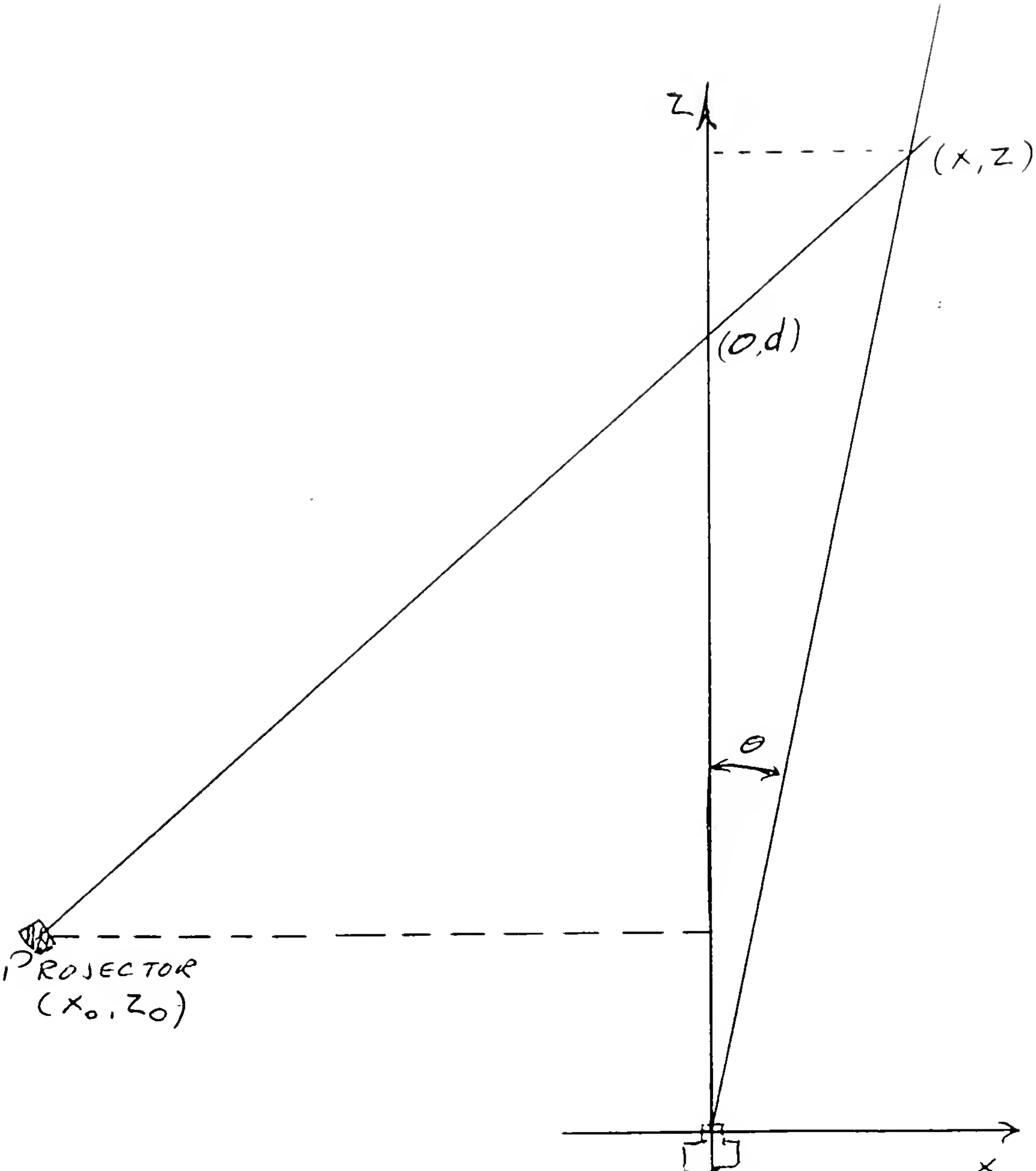


Fig. 2. Definition of quantities used in the derivation (see text).

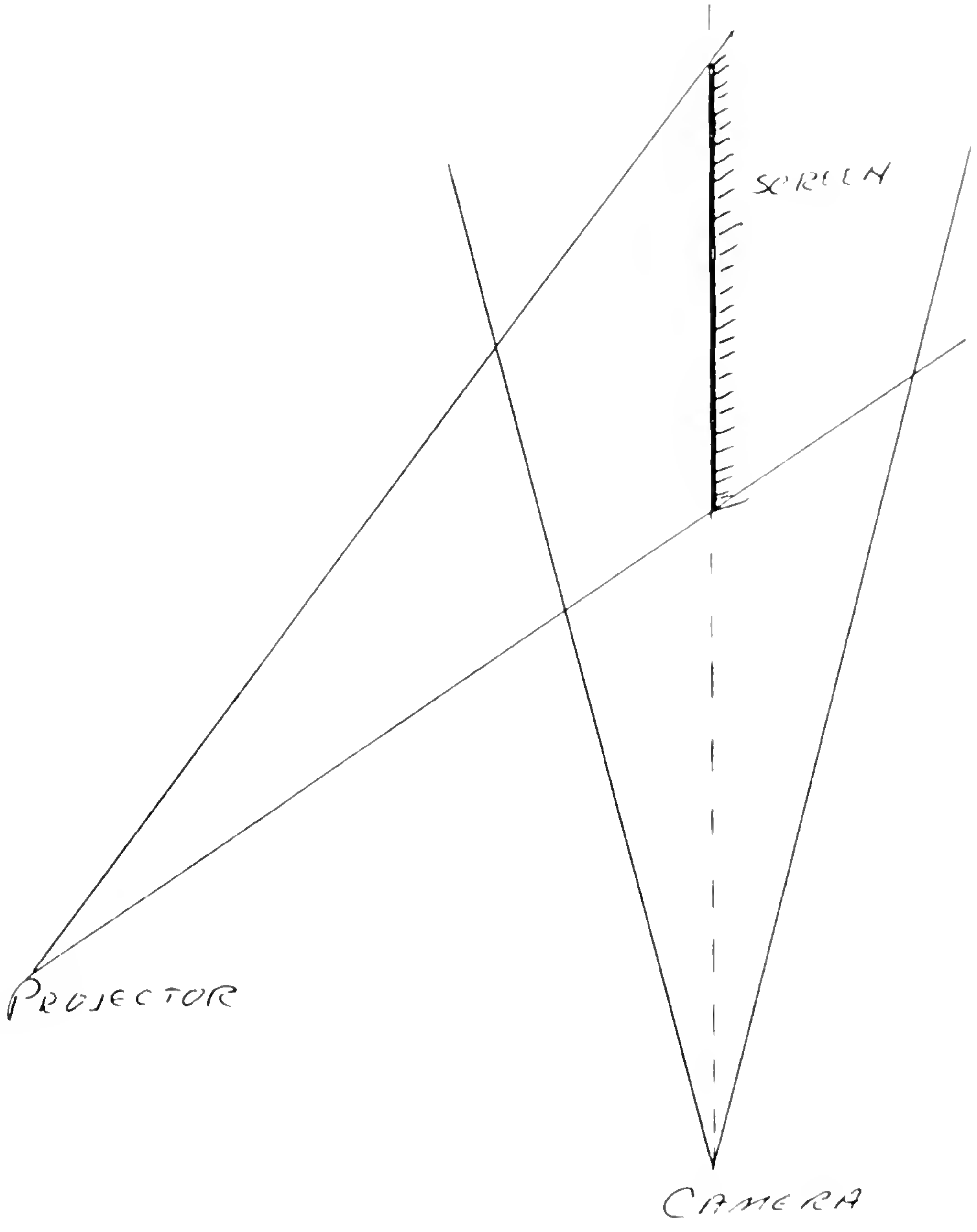


Fig. 3. Ratio values along this screen define $d(\rho)$.

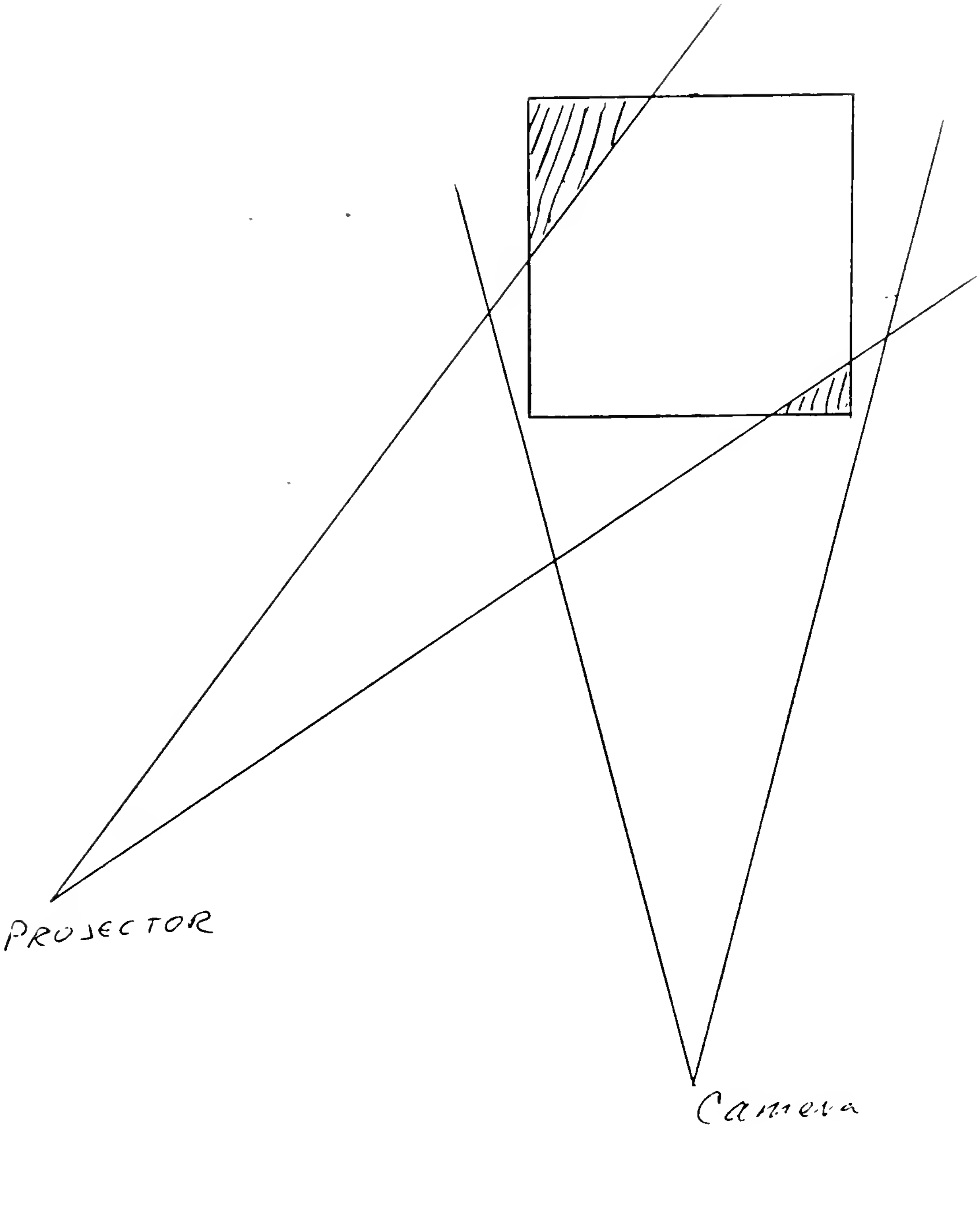


Fig. 4.. If measurements are made with projector in the calibration location some regions of the "workspace" are missing.

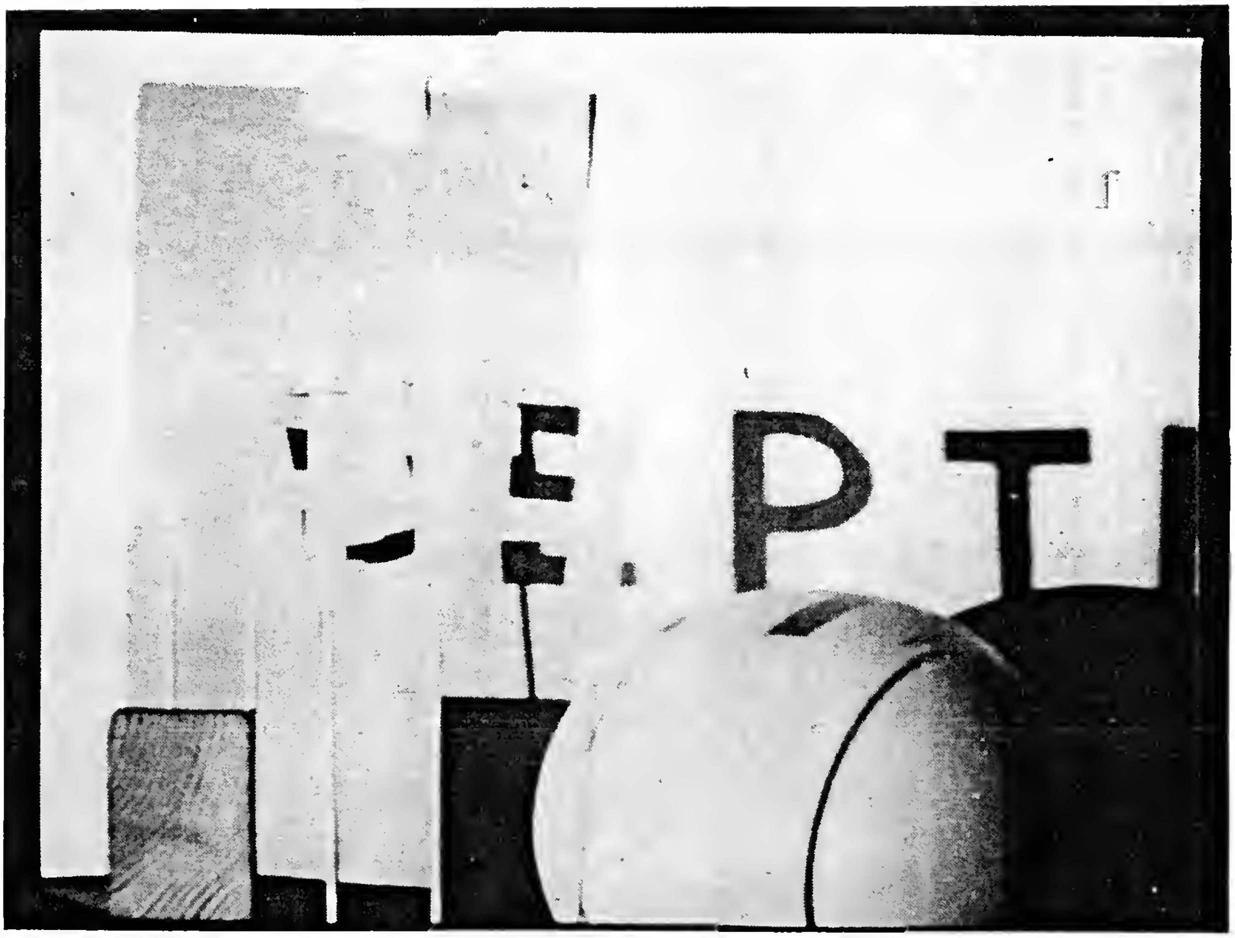


Fig. 5. Intensity image of the workspace as seen by the video camera.
Note the dark paper, and grain on the wood block.

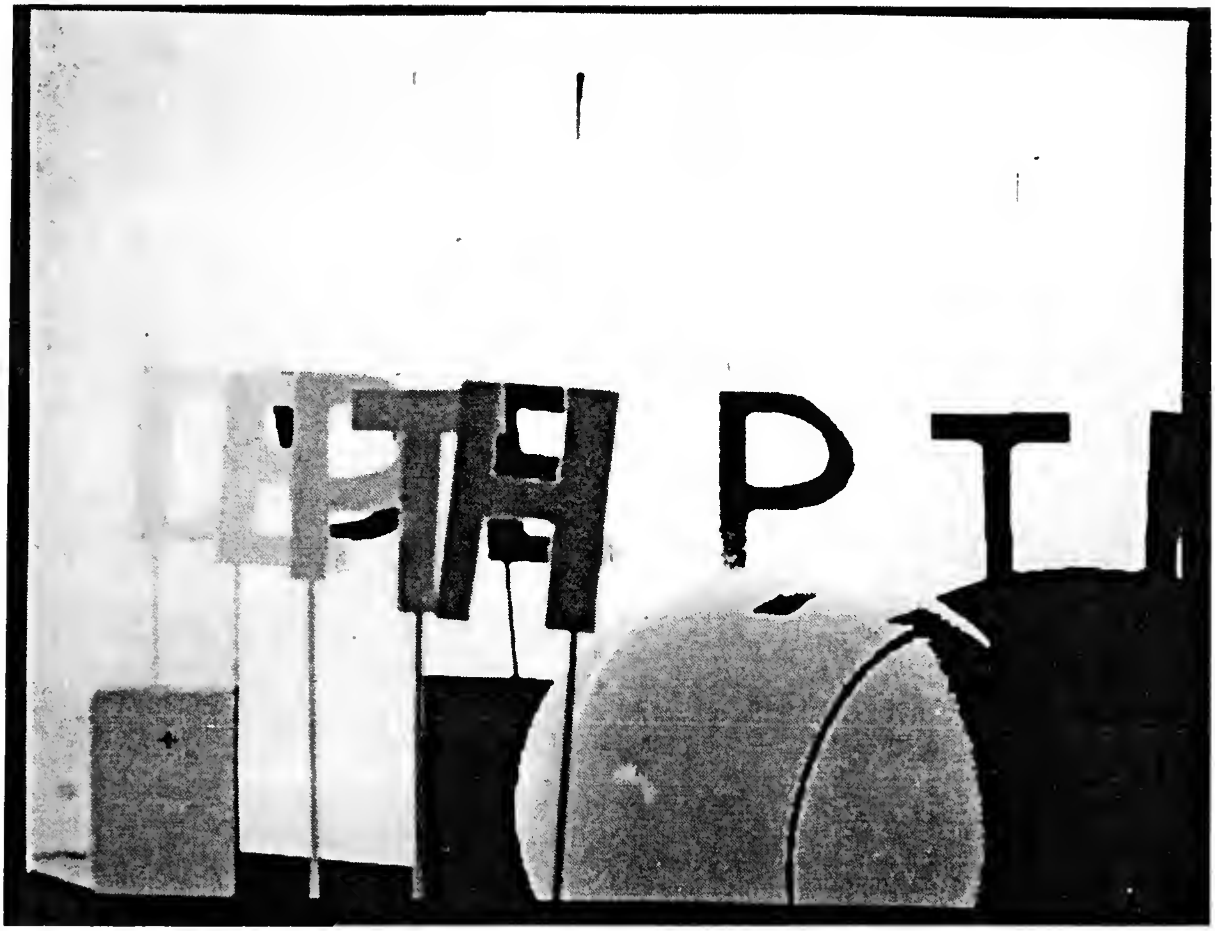


Fig. 6. Depth image of the workspace. Brighter values are farther from the camera. Shadows are tagged black (no depth information).

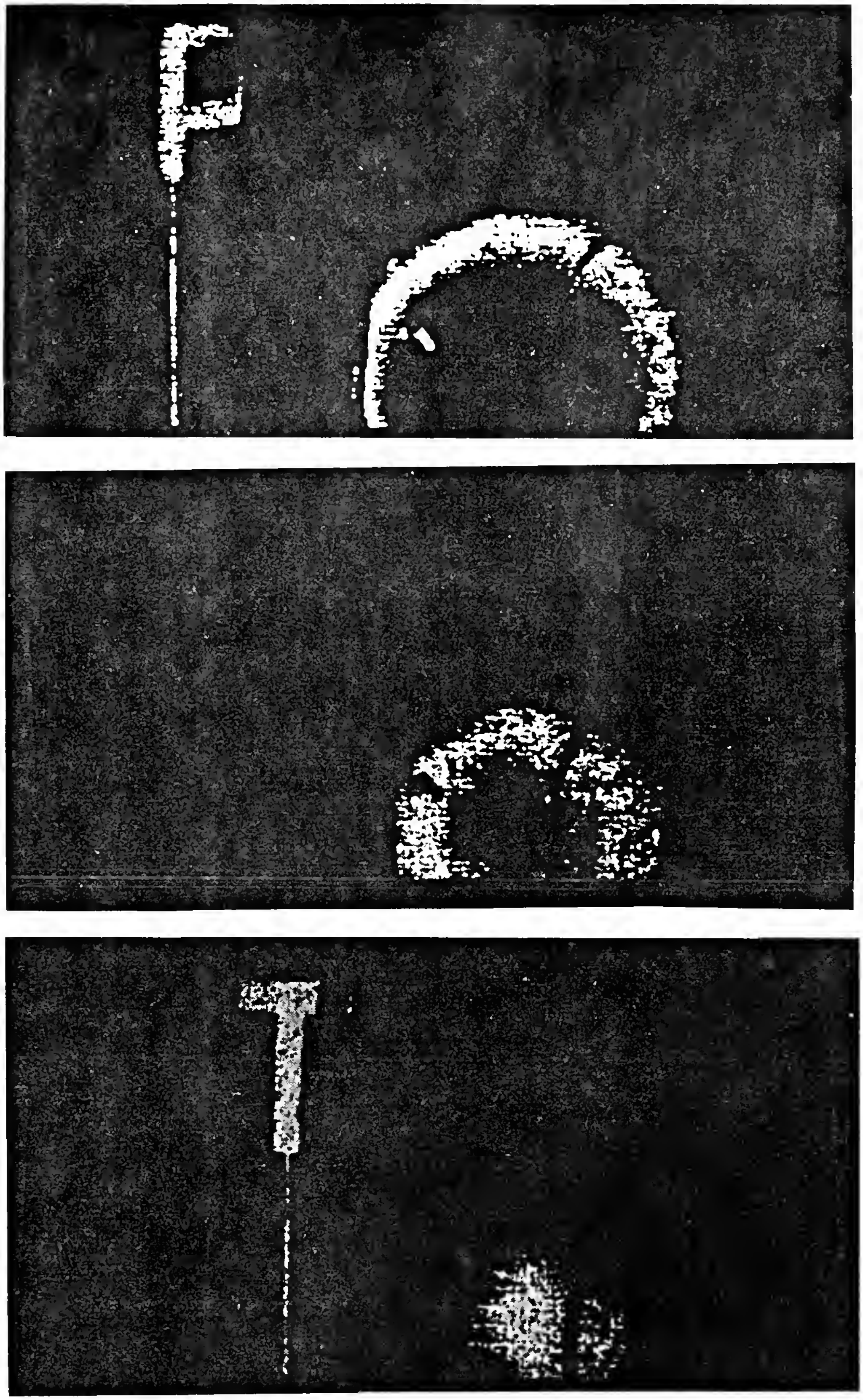


Fig. 7. Sequential 2 cm sections through the depth image. All portions of the image lying outside a section are black.

NYU COMPSCI TR-141
Bastuscheck , C Marc

Look-up table
compuation for ratio